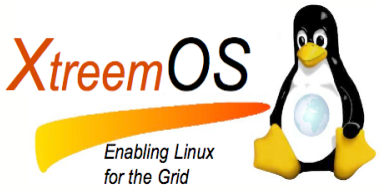


KERLABS

HEINRICH HEINE
UNIVERSITÄT
DÜSSELDORF

Incremental and SYSV IPC shm checkpointing

John Mehnert-Spahn
University of Duesseldorf
Germany

Information Society
Technologies

1.1 Incremental Checkpointing (ICP)

- **save modified content, only**
  - no code, libraries
- page granularity
  - (adaptive mechanisms exist)
- approach:
  - use WRITE bit of page table entries (PTE) to detect write accesses

1.2.1 Current ICP Implementation I

- cases:
  - initial cp (**full cp**)
  - non-initial cp (**increm. cp**)
- control structure – one entry per page
  - localize page data (which task_mm_pid.x.bin file)
  - entry: virtual address, file version, file offset
  - assemble entries in red black tree (rb tree)
- one control structure per process

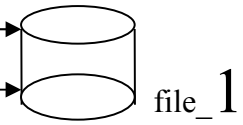
1.2.1 Current ICP Implementation II

- at checkpoint:
  - update entry content (using previous version of control structure – non-initial cp)
  - reset WRITE bit of relevant PTE
  - save entries into additional cp image file

1.2.2 ICP Control Structure Example

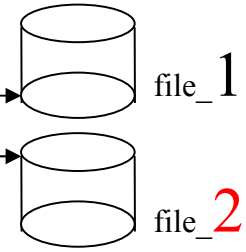
- initial checkpoint (cp version 1)

offset	virtual address	cp version
0	0x100	1
1	0x200	1



- incremental checkpoint (cp version 2)

offset	virtual address	cp version
0	0x100	1
0	0x200	**2**



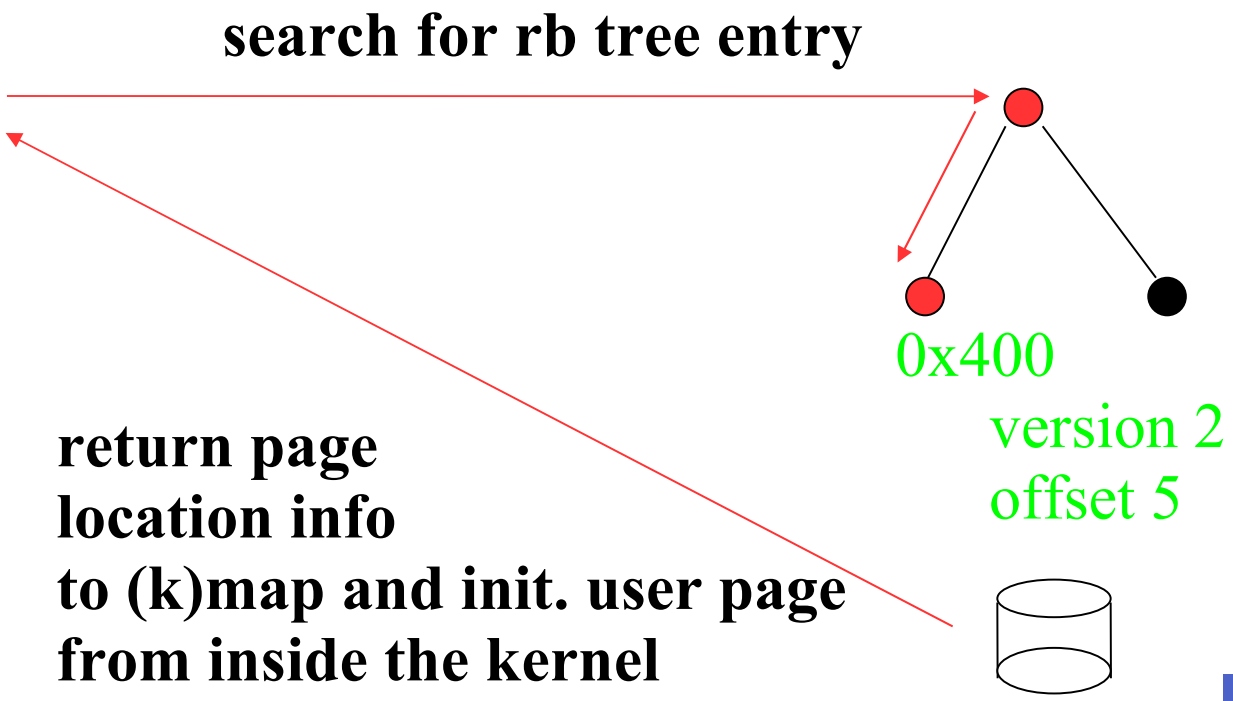
- misc_t – number VMA's and virtual addresses

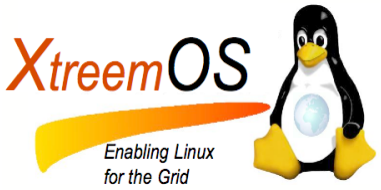
1.3 ICP-based Restart I

- localize physical page data for given virtual page address (task_mm file, offset)
  - assemble control structure entries in rb tree
  - virtual address as rb node key
  - tree accelerates search of approp. entry (before: list used)
- search appropriate tree entry for given virtual address
- retrieve page (and page prot. info) from appropriate task_mm file

1.3 ICP-based Restart II

mm_struct
VMA 1
vm_start (0x00)
...
0x400
...
vm_end
VMA 2
...





KERLABS

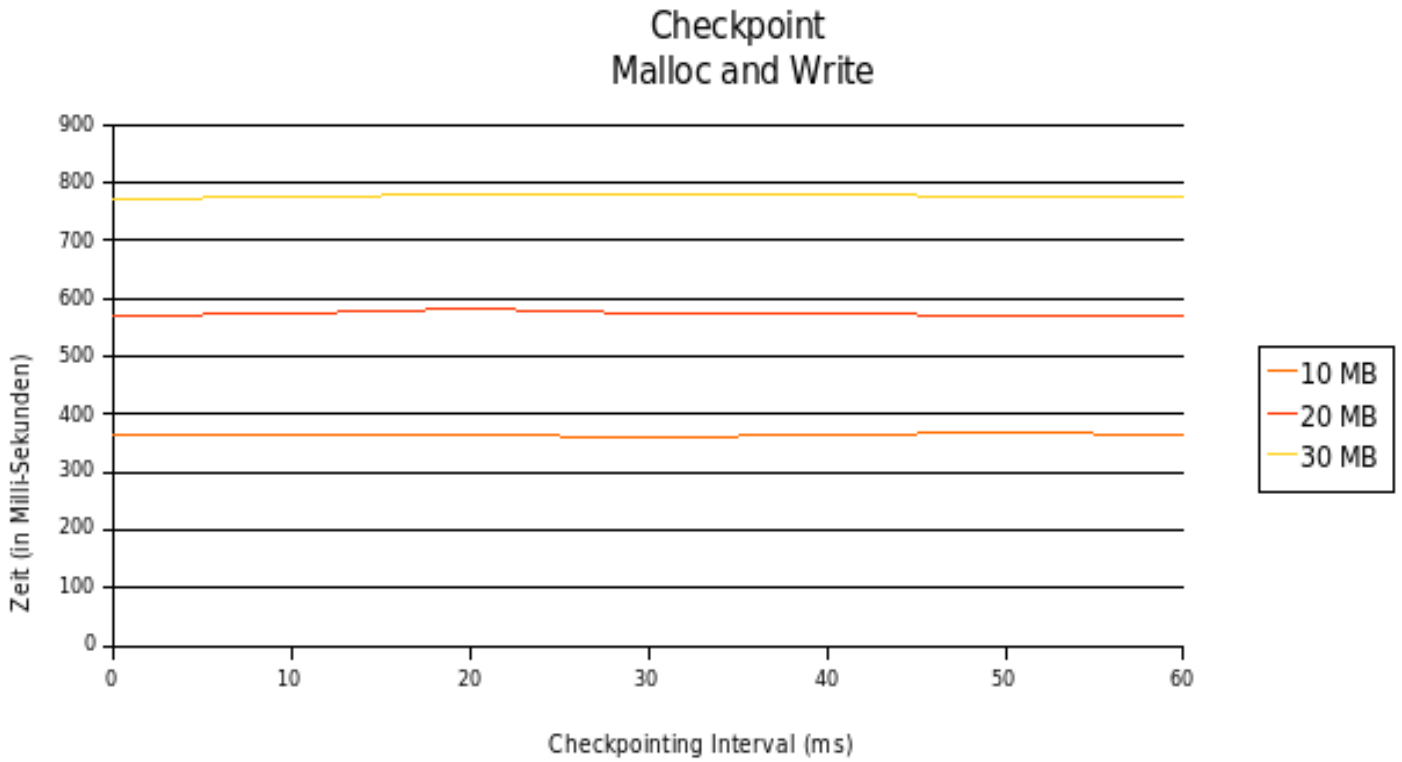
HEINRICH HEINE
UNIVERSITÄT
DÜSSELDORF

1.4 Optimization

- rb tree is not optimal for search operations → O(logn)
- alternative: avoid searching, just read in entries
  - control structure entries reflect virtual memory structure and provide physical page information (MMU)
  - O(1) access
  - page tables are allocated if needed, only

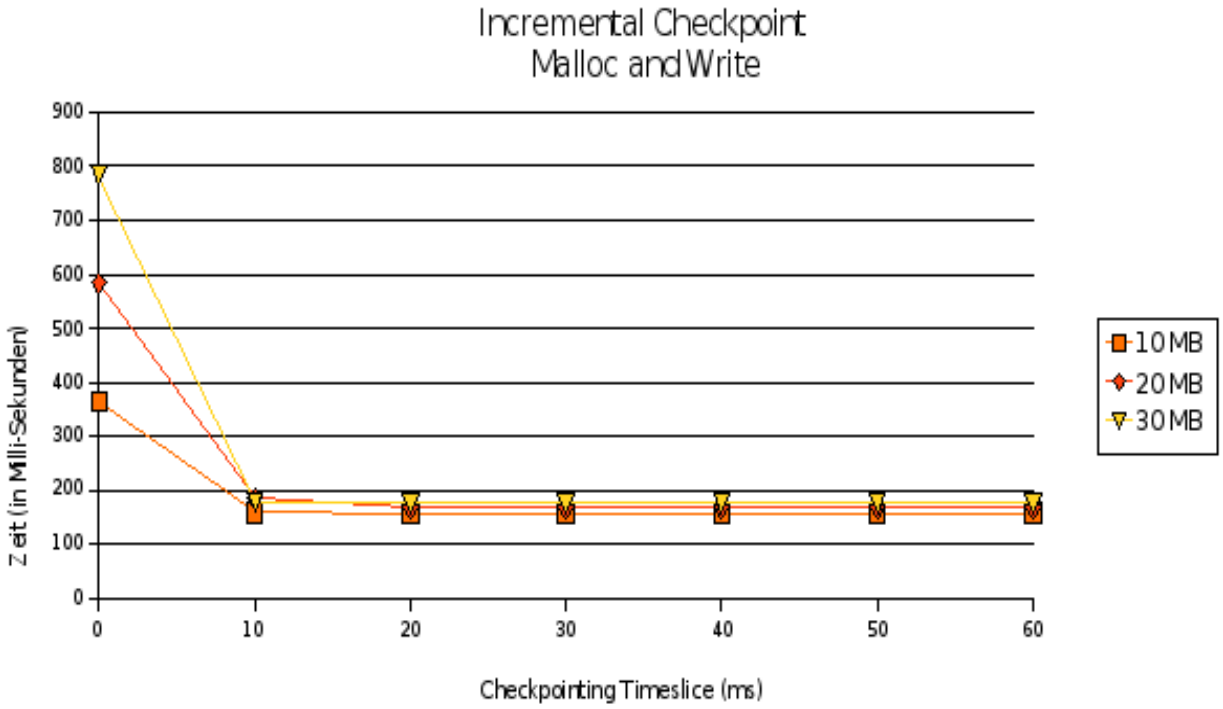
1.5.1 Measurements I

- malloc and write – **full cp** (cp time)



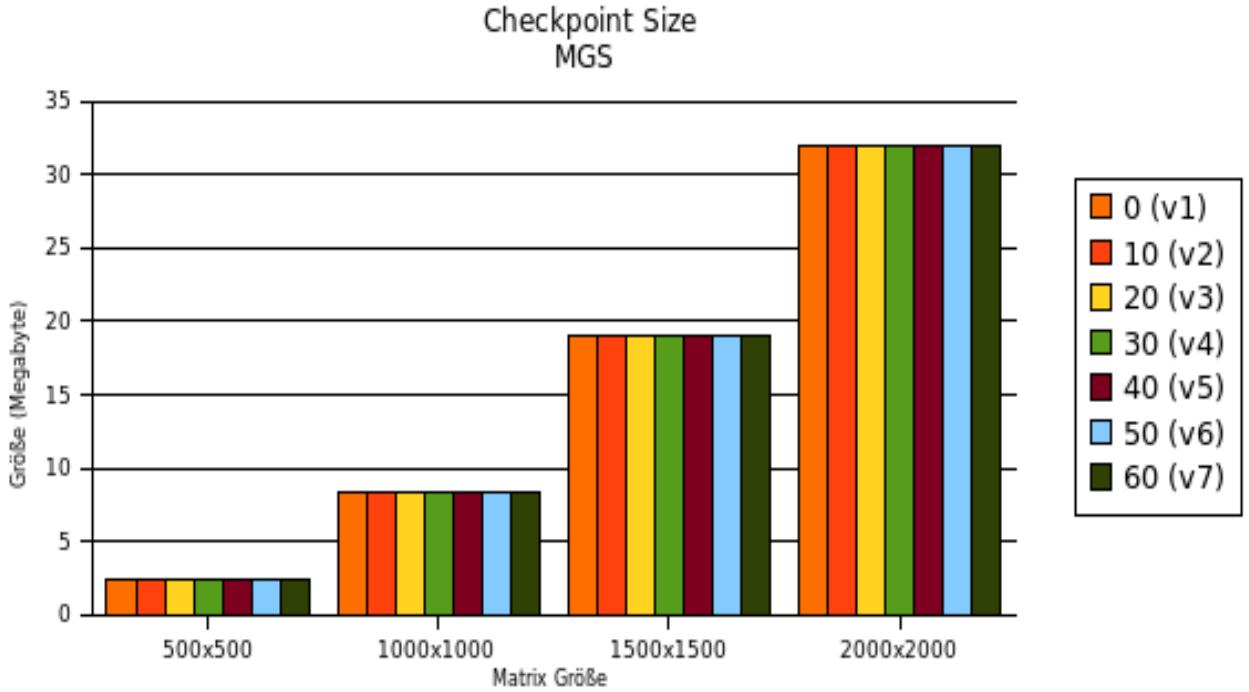
1.5.1 Measurements II

- malloc and write – **incremental cp** (cp time)



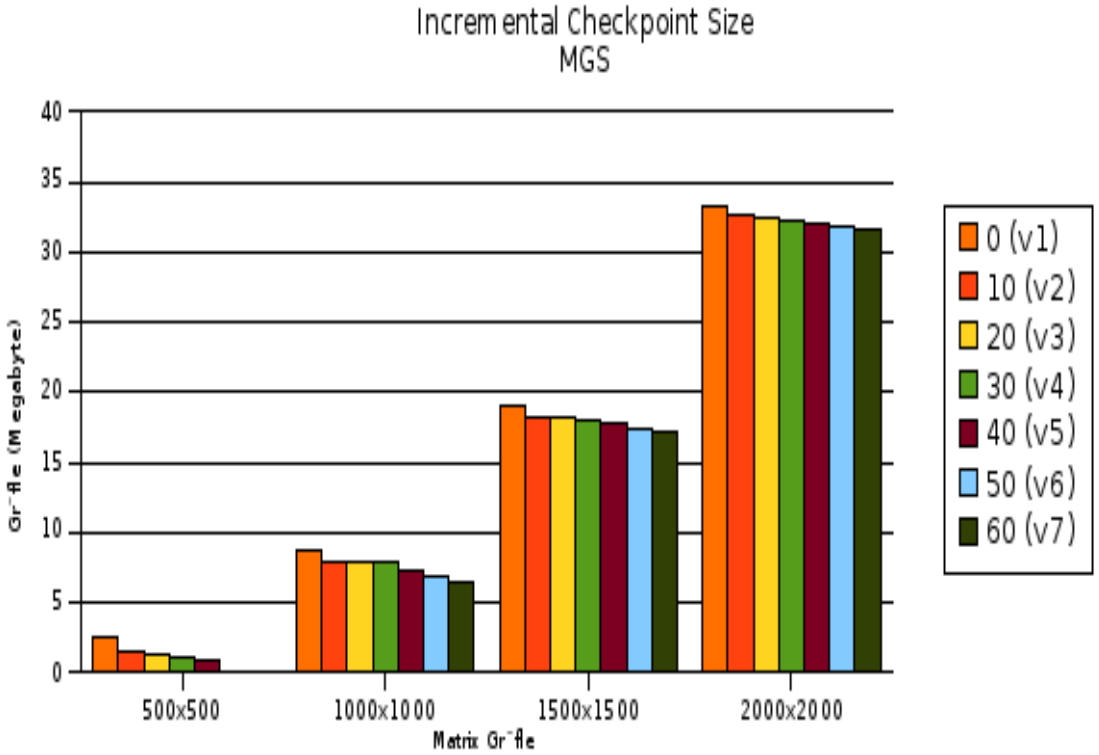
1.5.2 Measurements I

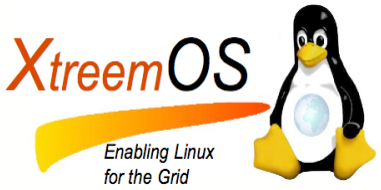
- MGS – **full checkpoint** (image size)



1.5.2 Measurements II

- MGS – **incremental checkpoint** (image size)





KERLABS

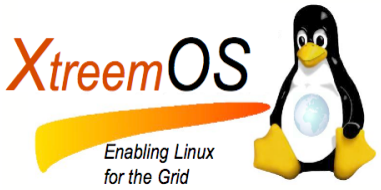
HEINRICH HEINE
UNIVERSITÄT
DÜSSELDORF

2.1 SYSV IPC shared memory basics

- often used for cluster applications
- mulitple processes can read from/write to a common memory area
- fastest IPC mechansim for process communication
- shm key identifies common segment
- proc interface for shm information
- memory resistency, IPC_RMID flag and `ipcrm`

2.2 SYSV IPC shm in Kerrighed

- support for local and distributed(!) applications
- system calls for typical shm application:
  - `shmget`
    - struct shmid_kernel
    - newseg call (shm file, shm kddm set)
  - `shmat`
    - vmops
    - shm_inc



KERLABS

HEINRICH HEINE
UNIVERSITÄT
DÜSSELDORF

2.3 Features of the cp/rst shm implementation

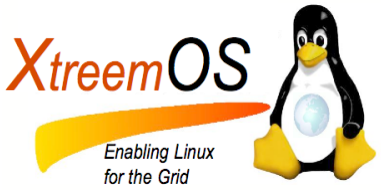
- support for parent-child (fork)
  - no duplicated rebuilding of structures
- support for changed roles at restart (orig. server becomes client, and vc.)
  - no dependency on original server
- support for distributed application
- (shm) namespace ID's not saved
  - at restart: use same shm key as before, recreate namespace ID's – might be different

2.4 Impl. of shm CP

- approach: emulate subset of system call's functionality
- saved data:
  - shm key
  - segment size
  - segment content
- unsaved data:
  - `shmkey, shmid, ipcmap` KDDM set objects
- impact on code
  - add file ipc_cr.c|h
  - extend `export_one_page`
  - export one kernel symbol (newseg)

2.5 Impl. of shm RST

- emulate subset of **`shmget`** and **`shmat`** functionality
- first process sets up segment
- second, third ... process
  - local – link to existing segment, multiple structure rebuilding avoided (no **`newseg`** call!)
  - remote – **`shmget`** cares
- impact on code:
  - extend `shm_object` KDDM object
  - `kcb_new_newseg`
  - `import_shm_file`
- no virtualization for shmid needed
  - just recreate, shmkey is the same

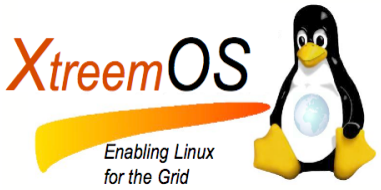


KERLABS

HEINRICH HEINE
UNIVERSITÄT
DÜSSELDORF

2.6 Misc.

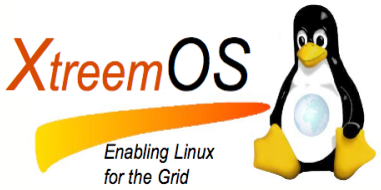
- when to call ipcrm?
  - XtreemOS: job concept (procs. involved are known)
  - Kerrighed: no such concept – user's task to call
- no knowledge about when which process will restart
  - existing segment content might differ with ghost content



KERLABS

HEINRICH HEINE
UNIVERSITÄT
DÜSSELDORF

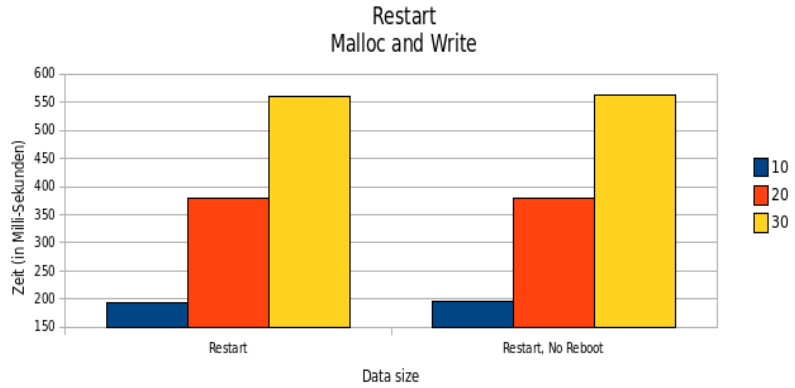
Thank you for your attention!



BACKUP

1.5.2 Measurements

- malloc_and_write – full restart (rst time)



- malloc_and_write – incr. restart (rst time)